International Journal of Networking and Computing – `www.ijnc.org`
ISSN 2185-2839 (print) ISSN 2185-2847 (online)
Volume 5, Number 1, pages 223–238, January 2015

An Adaptive Routing Algorithm of 2-D Torus Network Based on Turn Model: The Communication Performance

Yasuyuki Miura, Kentaro Shimozono, Naohisa Fukase, and Shigeyoshi Watanabe
Department of Information Science, Shonan Institute of Technology
1-1-25, Tsujido Nishikaigan,Fujisawa, Kanagawa, Japan

Kazuya Matoyama
Net-Research Co.Ltd
1758-7, Shimo-Numabe, Nakahawa, Kawasaki, Japan

Received: February 11, 2014
Revised: May 6, 2014
Revised: July 7, 2014
Accepted: July 29, 2014
Communicated by Shuichi Ichikawa and Takashi Yokota

Abstract

A 2-D torus network is one of the most popular networks for parallel processing. Many algorithms have been proposed based on the turn model, but most of them cannot be applied to a torus network without modification. In this paper, we propose North-South First (NSF) routing that is applicable to a 2-D torus and combines the north-first method (NF) and the south-first method (SF). NF and SF are algorithms yielded by the turn model. A software simulation comparing NSF routing with other forms of deterministic and adaptive routing showed that NSF routing improves throughput in three types of communication patterns, but yields no improvement for one other communication pattern.

Keywords: Network on Chip, Interconnection Network, Adaptive Routing, Turn Model

1 Introduction

The interconnection network is an important topic in the field of parallel processing. Parallel computers have processing elements (PEs) that are directly connected to a network such as a k-ary n-cube. Parallel processing is also performed in a *Network on Chip (NoC)* between PEs located on one chip. Many different interconnection networks for parallel processing have been proposed, and the 2-D torus network is one of the most popular networks for parallel processing.

The routing algorithms of interconnection networks are classified into deterministic routing, in which paths are fixed, and adaptive routing [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11], in which paths are changed to avoid failures or congestion. Because of its tolerance to failures and congestion, adaptive routing has been the topic of a lot of researches. Various adaptive routing algorithms have been proposed for k-ary n-cubes [6, 7, 8, 9, 10, 12, 13]. However, these methods require additional hardware for virtual channels. For example, Dally et al. proposed two types (dynamic and static) of *dimension reversal routing*[7]. Those methods require at least one (dynamic) or two (static) additional virtual channel for achieving adaptive routing. Also, the fully adaptive routing for k-ary n-cubes based on Duato's method [11] is possible. But this method requires one additional virtual channel for bypass path for achieving adaptive routing.

A number of adaptive routing algorithms based on the turn model [14, 15, 16, 17] do not need additional virtual channels. However, most of these algorithms cannot be applied to torus networks without change. If an adaptive routing algorithm for a torus network could be realized by modifying the turn model, it would be possible to realize adaptive routing without having to install additional virtual channels [18].

In our previous researches, we proposed restricted North First (rNF) [19]. But the improvement of performance was not obtained in this method. So we proposed to combine the South First (SF) method to rNF [20]. This method was named the *North-South First* (NSF) routing (SF is adaptive routing algorithms based on turn model).

In [20], we presented the NSF Routing algorithm and evaluated its dynamic performance in a software simulation of two communication patterns. The simulation, however, only compared the NSF method with the conventional *deterministic routing* (its name is *Dimension Order Routing*, DOR) on a 2-D torus. And the communication patterns were limited.

In this paper, we evaluate the communication performance of the NSF method on a 2-D torus in comparison with a number of conventional methods for 2-D tori and mesh networks. Moreover, we evaluated two more patterns in addition to the two patterns covered in the previous research.

2 2-D Torus Network

The structure of a 2-D torus network is shown in Fig.1. The network has an $N \times N$ 2-dimensional structure, and its four edges are connected by wraparound links. It is used in many parallel computers and some interconnection networks include this.

Dimension order routing (DOR) is generally used for deterministic routing on a 2-D torus. In DOR, the packet moves on channels in the y-direction before moving to the x-direction. To avoid deadlocks on a 2-D torus, DOR needs two virtual channels (channel-L and channel-H).

The method of selecting a virtual channel in the case of DOR on a 2-D torus network is as follows:

- Choose channel-L when starting routing in the y-direction.
- When the head of the packet passes through a wraparound link, move the packet to channel-H.
- When the routing in the y-direction is completed, move the packet in the x-direction; use channel-L regardless of the current channel.
- When the head of a packet passes through a wraparound link, moves the packet to channel-H. Use channel-H until the routing finishes.

Figs. 2 and 3 show the link selection function and channel selection function of DOR on an $N \times N$ torus. Here, the address of each PE of the torus is shown in terms of their coordinates (x, y). Moreover, the y-direction channels are written as Y+ and Y−, and the x-direction channels are written as X+ and X−. The four inputs of the link selection function indicate the x and y coordinates of the present PE, and the x and y coordinates of the destination PE. The function outputs the link of either X+, X−, Y+, Y− or "OUT", which is an output link to a node.

The three inputs of the channel selection function correspond to the current direction, current channel, and direction of the next hop. The current direction and the direction of the next hop have four states, i.e., X+, X−, Y+, and Y−. The current channel has three states, i.e., channel-L (L), channel-H (H), and wraparound channel (W). Although the output has two states (L and H), it unconditionally serves as W when the selected link is a wraparound link.

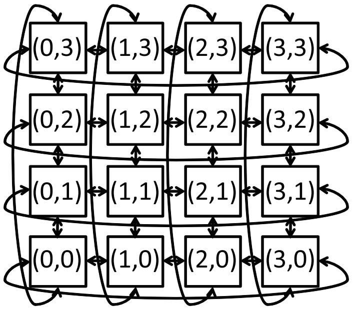
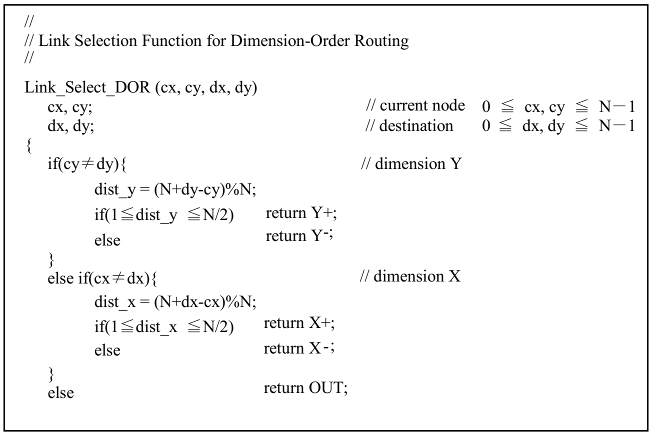


Figure 1: Structure of 2-D Torus Network

.

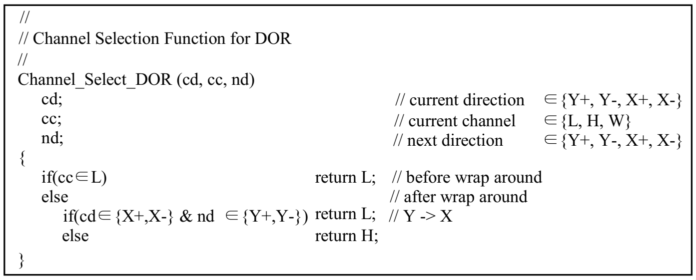


```
//
// Link Selection Function for Dimension-Order Routing
//

Link_Select_DOR (cx, cy, dx, dy)
    cx, cy;                                   // current node   0 ≦ cx, cy ≦ N−1
    dx, dy;                                   // destination    0 ≦ dx, dy ≦ N−1
{
    if(cy≠dy){                                // dimension Y
            dist_y = (N+dy-cy)%N;
            if(1≦dist_y ≦N/2)     return Y+;
            else                  return Y-;
    }
    else if(cx≠dx){                           // dimension X
            dist_x = (N+dx-cx)%N;
            if(1≦dist_x ≦N/2)     return X+;
            else                  return X-;
    }
    else                          return OUT;
```

Figure 2: Link Selection Function of Dimension-Order Routing

.



```
//
// Channel Selection Function for DOR
//
Channel_Select_DOR (cd, cc, nd)
    cd;                                               // current direction   ∈{Y+, Y-, X+, X-}
    cc;                                               // current channel     ∈{L, H, W}
    nd;                                               // next direction      ∈{Y+, Y-, X+, X-}
{
    if(cc∈L)                               return L;  // before wrap around
    else                                              // after wrap around
       if(cd∈{X+,X-} & nd ∈{Y+,Y-})        return L;  // Y -> X
       else                                return H;
}
```

Figure 3: Channel Selection Function of Dimension-Order Routing

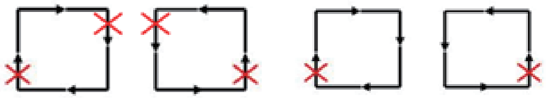
.

3 Adaptive Routing of k-ary n-cube

3.1 Turn Model

The turn model [15] is used by some adaptive routing algorithms [16, 17]. Packet cycles can be prevented by adding a restriction to a path change (turn) of a packet. In the case of a 2-D mesh, there are eight kinds of turn, and the various turn model methods put restrictions on two of the eight turns. There is essentially no difference between these methods other than the choice of turn to be restricted. In this paper, we shall incorporate the North First (NF) algorithm and South First (SF) algorithm into one (NSF) and apply it to a 2-D torus.

The turn model of DOR for a 2-D mesh is shown in Fig.4 a), and the turn model of the NF algorithm is shown in Fig.4 b). DOR restricts four out of eight turns, whereas the NF algorithm restricts only two, i.e., X− (left, west) → Y+ (upper, north) and the X+ (right, east) → Y+ (upper, north). The South First algorithm, by which the Y− (South) direction is chosen at the beginning of a routing path, is similar.



a) Dimension Order Routing b) North First Algorithm

Figure 4: Turn Model for 2-D Mesh Network

.

3.2 Application of the Turn Model to a Torus Network

When applying a turn model such as the NF algorithm to a torus network without change, the following differences from the case of a mesh network have to be considered.

1. In a torus network, when the packet passes through a wraparound channel, a deadlock by cyclic dependency can occur. Therefore, it is necessary to impose an additional restriction.
2. At least two virtual channels are needed for routing in a torus network. As a result, adaptive routing with higher pliability is attained by applying different turn models to each channel.

An example of a cyclic dependency that occurs in the NF algorithm is shown in Fig.5. Here, packets A-D mutually block a path, causing a deadlock. By contrast, the deadlock does not happen in DOR because packets A and C do not turn in Fig.5. This problem illustrates that it is necessary to take into consideration complicated turn restrictions in adaptive routing on a torus network. Our method deals with this issue by applying the NF and SF algorithms to channel-H and channel-L.

4 North-South First Routing

To avoid the sort of deadlock described above, additional restrictions have to be put on the NF and SF algorithms:

1. The SF algorithm does its routing on channel-H. However, a cycle may occur when a path is chosen in which a packet returns to channel-L through channel-H, and for this reason, DOR is carried out instead of the adaptive routing. In DOR, the x-direction channel chosen after a vertical (y-direction) wraparound channel has to be channel-L.

2. The NF algorithm does its routing on channel-L. Because the path of channel-H → channel-L exists after a wraparound channel, the cycle shown in Fig.5 occurs. As shown in Fig.6, though, the cycle can be avoided by adding one more restriction to the other two. Here, three restrictions are put on eight turns, specifically, right → upper, left → upper, and right → lower. This algorithm was named *restricted North First* (rNF) [20].

4.1 Definitions

From here on, all channels will be described in terms of their dimension $d \in \{X,Y\}$, direction $\delta \in \{+,-\}$, channel type $c \in \{L,W,H\}$, i.e., $(d\delta, c)$. X means X dimension, Y means Y dimension, and L, W, and H means channel-L, wraparound channel, and channel-H. $(d+, c)$ and $(d-, c)$ will be shown as a set, written as $(d\pm, c)$.

4.2 Routing Algorithm

In our method, the restricted NF algorithm is carried out in channel-L and the SF algorithm is carried out in channel-H. Since (Y−,L) and (Y+,H) are respectively used in the restricted NF algorithm and SF algorithm, we will study cases in which (Y+,c) is used and not used, and cases in which the horizontal and vertical wraparound channels are used and not used.

Figs. 7 and 8 show the link selection function and channel selection function of the proposed method on a $N \times N$ torus. As in the case of DOR in Fig.2, the link selection function outputs X+, X−, Y+, Y−, or “OUT” (an output link to a node). The proposed method needs the “current channel” as an input in addition to the inputs of DOR.

The channel selection policy varies depending on whether (Y+,c) is used or not. If it is used, adaptive routing is carried out only when the wraparound channels is not to be used from that point on. If (Y+,c) is not used, the restricted NF algorithm is carried out from the source PE until the first wraparound channel (or destination PE) is reached.

The algorithm of Fig.7, in ①, first determines whether the wraparound links of X and Y are used. In this case, the determination is based on the X and Y coordinates of the source and destination PEs as follows:

- When the difference between the X coordinates of the current PE and destination PE is less than $N/2$, h_wrap is set to 0 because the wraparound channel of the x-direction is not straddled. If not, h_wrap is set to 1.
- When the difference between the Y coordinates of the current PE and destination PE is less than N/2, v_wrap is set to 0. If not, v_wrap is set as 1.

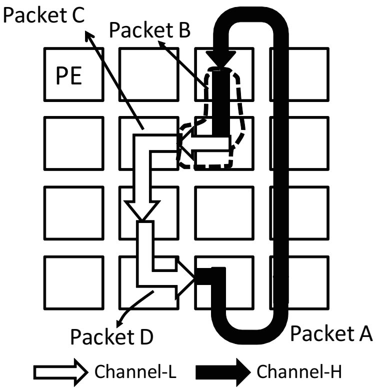


Figure 5: Cyclic Dependency Occurring in the NF Algorithm Running on a Torus Network

.

Next, the link is chosen on the basis of whether the Y+ channel (channel (Y+,c)) is used or not, as follows:

- When (Y+,c) is used, the procedure ② is carried out. In this case, since the restricted NF in channel-L is equivalent to DOR, only the adaptive routing of the SF method in channel-H is carried out. If neither wraparound channel is used in going from the current PE to the destination, the packet can be sent over channel-H and routing can be continued. Thus, adaptive routing can be carried out with the SF method. The only other case in which channel-H may be used is after the packet has passed through a vertical wraparound channel (Y+, W) and is due to pass through a horizontal wraparound channel (X±, W). Even in this case, it is thought that adaptative routing using the SF method is possible. However, since it is difficult to prove that is deadlock-free, only the X-directional routing is carried out from (Y+, W) to (X±, W) and SF is applied after the packet has passed through (X±, W). So, the load concentrates in part of the network (near PE$(0, 0)$).

- When (Y+,c) is not used, the procedure ③ is carried out. Since the SF method in channel-H is equivalent to DOR, only the adaptive routing of the restricted NF method in channel-L is carried out. In this case, the following restriction is added in order to make the order of passage in a wraparound channel into (Y−, W) → (X±, W).
  - Restricted NF is carried out only when (Y−, W) is not be passed from the current PE to the destination or the next channel is not (X±, W). DOR is carried out otherwise.

Besides the three inputs of the channel selection function of DOR in Fig.3, the channel selection function needs four inputs that indicate the x and y coordinates of the source and destination PEs. These new inputs can be used to judge the possibility of the packet passing through a wraparound channel. Based on the judgment, channel-H is chosen only when the wraparound channel is not to be used and (Y+, c) is to be used. DOR is carried out otherwise. As in the case of DOR, the output has two states, L and H. However, an output unconditionally serves as W when the selected link is a wraparound link.

With this algorithm, the load concentration may happen in the following cases;

- When (Y+,c) is not used ((Y-,c) is used) and a lower-left node sends a packet to an upper-right node through both of the wraparound channels, the restricted NF (where the packet goes left (X−) in this case) is applied until the packet arrives at the node $(0, y)$, and then the DOR is carried out. Therefore such a packet has to pass the (Y−, W) channel of (0, 0). So the load concentrates at there. Such a thing may happen to the packets which use the both of (X±, W) and (Y−, W).

- When (Y+,c) is used, the load concentrates in part of the network, as mentioned in above. Such a thing happens to all the packets which use the both of (X±, W) and (Y+, W).

They means the algorithm has to be modified further for it to work in all cases as future work.

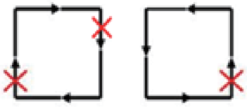


Figure 6: Restricted North First Routing

.

```
//
// Link Selection Function for Proposed Algorithm
//
Link_Select_Prop (cx, cy, cc, dx, dy)
    cx, cy;                                         // current node        0 ≦ cx, cy ≦ N−1
    cc;                                             // current channel     ∈{L, H, W}
    dx, dy;                                         // destination         0 ≦ dx, dy ≦ N−1
{
    if(|dx-cx|≧N/2)              h_wrap = 1;
    else                         h_wrap = 0;
    if(|dy-cy|≧N/2)              v_wrap = 1;            ①
    else                         v_wrap = 0;

    dist_y = (N+dy-cy)%N;
    if(1≦dist_y ≦N/2)                               // Y+ direction
        if(h_wrap=0 & v_wrap=0)
                                 return adaptive_SF(cx, dx);
        else if(h_wrap=1 & v_wrap=0)
                                 return x_route(cx, dx);    ②
        else                     return DOR(cx, cy, dx, dy);
    else if(cy≠dy)                                  // Y- direction
        if((cc∈L) and ((cx≠0) or (v_wrap=0))
                                 return adaptive_NF(cx, dx);
        else                     return DOR(cx, cy, dx, dy);    ③
    else if(cx≠dx)               return x_route(cx, dx);
    else return OUT;
}

adaptive_SF(cx, dx){                                //adaptive routing of SF algorithm
    if(cx=dx)                    return Y+;
    else if(buffer_is_full(Y+, H)=TRUE)
                                 return x_route(cx, dx);
    else                         return Y+;
}

adaptive_NF(cx, dx){                                //adaptive routing of rNFalgorithm
    dist_x = (N+dx-cx)%N;
    if(cx=dx)                    return Y-;
    else if(N/2≧dist_x)                             // X+direction
                                 return X+;
    else if(buffer_is_full(Y-, L)=TRUE)             // X- direction
                                 return X-;
    else                         return Y-;
}

x_route(cx, dx){
    dist_x = (N+dx-cx)%N;
    if(1≦dist_x ≦N/2)            return X+;
    else                         return X-;
}

DOR (cx, cy, dx, dy){
                                 return Link_Select_DOR (cx, cy, dx, dy);
}
```

Figure 7: Link Selection Function of the Proposed Algorithm

.

4.3 Deadlock Avoidance

A channel dependency graph is drawn in order to prove that the routing algorithm described in the previous section does not cause a deadlock [11, 21, 22]. The channel dependency graph is a directed graph in which nodes (channels) with dependencies are connected by an arrow. Specifically, nodes (channels) with dependencies are pairs of nodes (channels) in which a packet may be directly transmitted and received while routing.

First, the channel dependency graph is drawn. Then, each channel is numbered. If it is proved that the channel numbers are in ascending order (or descending order) in the direction of the arrows of the channel dependency graph, deadlock does not happen. In such case, the channels are said to have an ordered relation and the corresponding channel will not cause a cyclic dependency.

A routing algorithm based on the turn model generally assigns numbers to the output channels from the PE on the basis of the PE address. As mentioned above, a 2-D torus network has two virtual channels. Accordingly, the following 4-dimensional channel numbers CN are given to the 4 links ×2 channels (= 8 channels) in each PE of an $N \times N$ torus.

$$CN(x, y, d, ch) = (g_m, c_1, g_s, c_2) \tag{1}$$

Here, $x(0 \leq x \leq N-1)$ and $y(0 \leq y \leq N-1)$ are the x- and y-coordinates of the PE address, $d \in \{Y+, Y-, X+, X-\}$ is the direction of the channel, and $ch \in \{L, H, W\}$ is the type of channel. g_m,c_1,g_s, and c_2 denote the main group, first coordinate, sub group, and second coordinate, respectively. These values are numbered as follows:

- Main Group g_m

  The channel direction order is based on d and ch. Table 1 lists the value of g_m.

Table 1: Values of g_m determined by d and ch

d	ch	g_m
Y+	L, W	0
Y-, X-	L, W	1
Y-	H	1
X+	L,W	2
X+, X-, Y+	H	3

- First Coordinate c_1

  The value c_1 is based on g_m (see Table 2).

Table 2: Value of c_1 determined by g_m

g_m	c_1
0	y
1	$n-x$
2	0
3	y

- Sub Group g_s

  The sub group g_s value determines the order of channels in the same g_m. g_s is set to 0 at $g_m = 0$ and $g_m = 2$. Table 3 lists the values of g_s at $g_m = 1$ and $g_m = 3$.

- Second Coordinate c_2

  The second coordinate c_2 determines the order of the same d and ch. The c_2 values of y, $N-y$, x, and $N-x$ correspond to d values of Y+,Y−,X+, and X−.

Table 3: Value of g_s

g_m	d	ch	g_s
1	Y-	L, W	0
1	Y-	H	1
1	X-	L, W	2
3	X-, X-	H	0
3	Y+	H	1

Fig.9 illustrates the channel numbers for each channel. Here, deadlocks can be avoided because channel numbers will be in ascending order through a routing path. Fig.10 illustrates the channel numbers for a 3×3 torus network . Here as well, the channel numbers are in ascending order.

1. The following procedure is executed. If a wraparound channel is not used:

   I-a) If channel (Y+,c) is used, the SF method is carried out on channel-H. In this case, the value of c_1 increases when the packet passes through (Y+,H), and the value of c_2 increases when it passes through (X±,H). Thus, the channel numbers are in ascending order.

   I-b) If channel (Y−,c) is used, the rNF method is carried out on channel-L first. When the packet goes to the lower left (south-west), adaptive routing is carried out. In this case, the value of c_1 increases when it goes to the left ((X−,L) is used), and the value of c_2 increases when it goes through the lower (South) output ((Y−,L) is used). When the packet goes to the lower right, DOR is carried out. In this case, c_1 and c_2 respectively increase when the packet goes through the right and lower (South) outputs. Thus, the channel numbers are in ascending order.

2. If only channel (Y±, W) is used:

   II-a) If channel (Y+, c) is used, DOR is carried out before the packet passes through the wraparound channel. SF is carried out in the same way as in I-a) after the packet exits the wraparound channel. The channel numbers are in ascending order.

   II-b) If channel (Y−, c) is used, rNF is carried out before the packet passes through the wraparound channel. In this case, the channel numbers are in ascending order the same as in I-b). DOR is carried out after the packet has passed through the wraparound channel. The channel numbers are in ascending order.

3. When only the channel (X±, W) is used:

   III-a) If channel (Y+,c) is used, only DOR is carried out virtually. In this case, the channel numbers are in ascending order.

   III-b) If channel (Y−,c) is used, rNF is carried out before the packet passes through the wraparound channel and DOR is carried out after it has passed through, just as in II-b). The channel numbers are in ascending order like that of II-b).

4. When both wraparound channels are used:

   IV-a) If channel (Y+,c) is used, because DOR is carried out first on channel-L, the order of the wraparound channels along the route becomes vertical → horizontal. After passing through the vertical wraparound channel, the packet is routed through the horizontal channel ((X±, L)) (see the 4th line of Fig.7 ②). In this case, the channel numbers are in ascending order. Adaptive routing on channel-H is carried out after the packet passes through the horizontal wraparound channel. In this case, the channel numbers are in ascending order.

```
//
// Channel Selection Function for Proposed Algorithm
//
Channel_Select (cx, cy, dx, dy , cd, cc, nd)
    cx, cy;                                        // current node        0 ≦ cx, cy ≦ N−1
    dx, dy;                                        // destination         0 ≦ dx, dy ≦ N−1
    cd;                                            // current direction   ∈{Y+, Y-, X+, X-}
    cc;                                            // current channel     ∈{L, H, W}
    nd;                                            // next direction      ∈{Y+, Y-, X+, X-}
{
    if(dx-cx≧N/2)              h_wrap = 1;
    else                       h_wrap = 0;
    if(dy-cy≧N/2)              v_wrap = 1;
    else                       v_wrap = 0;

    dist_y = (N+dy-cy)%N;
    if((1≦dist_y  ≦N/2)                            // Y+ direction
            and (h_wrap=0 & v_wrap=0))
                               return H;
    else                                           // Others
                               return DOR_Channel (cd, cc, nd);
}

DOR_Channel (cd, cc, nd){
                               return Channel_Select_DOR (cd, cc, nd);
}
```

Figure 8: Channel Selection Function of the Proposed Algorithm

.

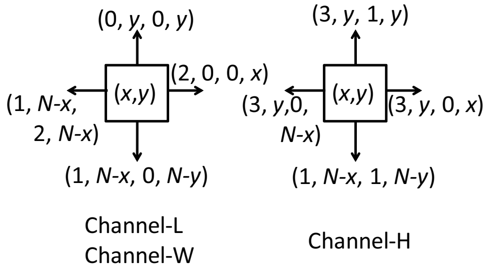
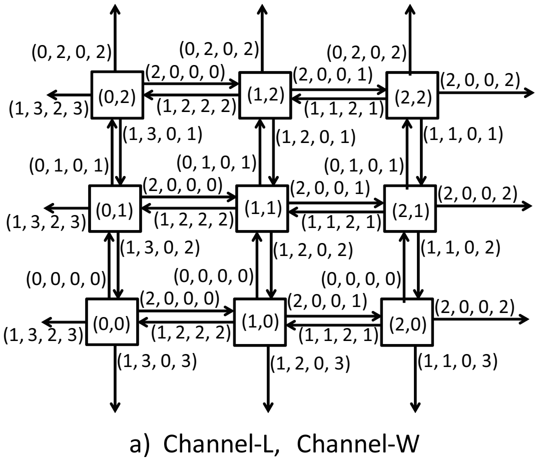


Figure 9: Channel Number

.



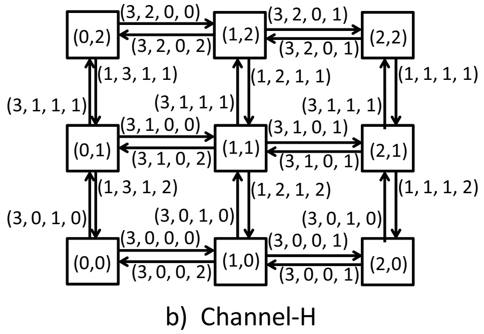


Figure 10: Channel Number of 3 × 3 Torus

.

IV-b) If channel (Y$-$,c) is used, adaptive routing on channel-L is carried out first. In this case, the channel numbers are in ascending order. The first line of Fig.7 ③ (the description of "cx$\neq$ 0") makes the order of the wraparound channels along the path become vertical →horizontal, because the packet going through channel-L cannot go to the horizontal wraparound channel before it goes to the vertical one. After the packet passes through the vertical wraparound channel, vertical channel routing on channel-H ((Y±,L)) is carried out. After it finishes, the routing path is equivalent to that of DOR because only horizontal routing is necessary. Thus, the channel numbers are in ascending order.

5 Performance Evaluation

5.1 Environment

We used a wormhole routing simulator to evaluate the dynamic communication performance of our algorithm on a 16×16 2-D torus/mesh network with 256 PEs. For comparison, we also simulated dimension-order routing (DOR) on a 2-D torus and DOR, north-last, west-first, and west-last on a 2-D mesh. The traffic patterns were uniform, matrix-transpose, bit-reversal, and longest-path.

The dynamic communication performance of an interconnection network was characterized by the average transfer time and throughput. The average transfer time was the average value of the latency for all packets. Latency was the time between the injection time of the first flit and the reception time of the last flit at the destination. Throughput was the average value of the number of flits which a PE receives in each clock cycle. In the evaluation of dynamic communication performance, flocks of messages were sent in the network so that they competed for the output channels. Packets were transmitted with a request probability r during T clock cycles and the number of flits which reached the destination PE and their transfer times were recorded. The average transfer time and throughput were then calculated and plotted. The request probability r was varied. The packet size was 16 flits, and flits were transmitted for 50,000 cycles, i.e., $T = 50000$. Two virtual channels per physical link were simulated. The buffer length of each channel was 8 flits.

5.2 Uniform Traffic

In this pattern, destinations were randomly chosen with equal probability among the nodes in the network. The results are shown in Fig.11. The throughput was not improved by adaptive routing on a mesh. This is because this pattern doesn't have any deviation. The throughput of the proposed method is slightly higher than that of DOR. In the uniform traffic communication pattern, the whole network is equally crowded. So the effect of avoiding crowded links is limited. However, in our method, since some packets are directly sent to channel-H, the load of that channel is distributed, and this is why the throughput is slightly higher.

5.3 Matrix-Transpose Pattern

The matrix-transpose traffic pattern is based on the transposition of a matrix. In this pattern, packets are transmitted between PEs over a diagonal line. We assumed that the number of PEs and data are the same. The elements of the matrix $A = \{a_{ij}\}$, a_{ij}, were assigned to corresponding PEs (i, j), and the communication of the transposition was carried out. Therefore, the traffic pattern of the matrix transpose consisted of communications between PE(i, j) and PE(j, i). The results are shown in Fig.12. Here, adaptive routing on a mesh had the higher throughput. This is because the matrix-transpose pattern has a deviation. The throughput of DOR was limited to 0.1, whereas that of the proposed method was 0.14. The ratio of progression of the performance with mesh and torus by adaptive routing were almost same.

5.4 Bit-Reversal Pattern

The bit-reversal traffic pattern targets the PE with the reverse address (reverse binary expression). Since the number of PEs was 256, this pattern consisted of communications from PE$(x, y) =$ PE$(x_3x_2x_1x_0, y_3y_2y_1y_0)$ to PE$(y_0y_1y_2y_3, x_0x_1x_2x_3)$. The results are shown in Fig.13. Here, adaptive routing on a mesh had the highest the throughput. The proposed method also improved the network load conditions.

5.5 Longest-Path Pattern

This traffic pattern has the maximum number of hops in a 2-D torus network. In an $N \times N$ torus, the packet moves $N/2$ hops horizontally and vertically. The packet from PE(i, j) moves to PE$(i + N/2 \bmod N, j + N/2 \bmod N)$. Here, $A \bmod N$ means "the remainder of A/N". In this case, a probability that a packet uses a wraparound channel is very high. The results are shown in Fig.14. The throughput was not improved by adaptive routing. This is because longest-path traffic doesn't have a deviation. The throughput of the proposed method was lower than the other methods. As mentioned in the last paragraph in 4.2, our current method concentrates the load at PE$(0, 0)$, and the performance decreases as a result. This problem remains to be resolved.

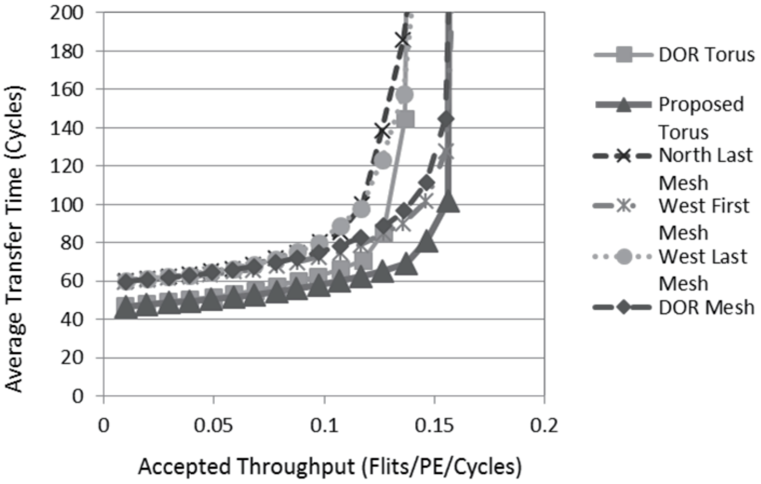


Figure 11: Results for Uniform Traffic

.

6 Conclusion

We proposed North-South First Routing (NSF Routing), which combines the North First method (NF) and South First method (SF). We evaluated its communication performance in a software simulation, and the results indicated that its throughput is higher those of other methods for some communication patterns. However, no performance improvement could be obtained for one communication pattern. We are planning to modify the NSF algorithm to prevent load convergence. We also plan to conduct a theoretical analysis of the flexibility of channel selection in order to evaluate the algorithm's fault tolerance.

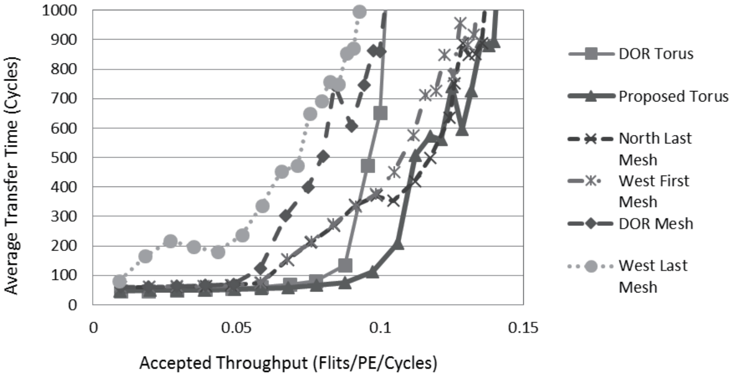


Figure 12: Results for Matrix-Transpose Pattern

.

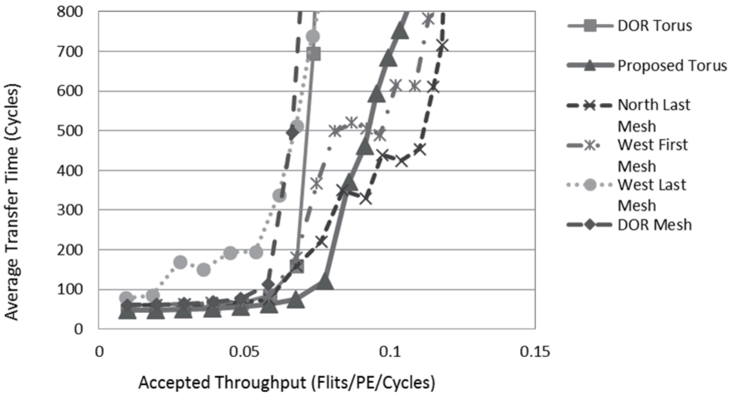


Figure 13: Results for Bit-Reversal Pattern

.

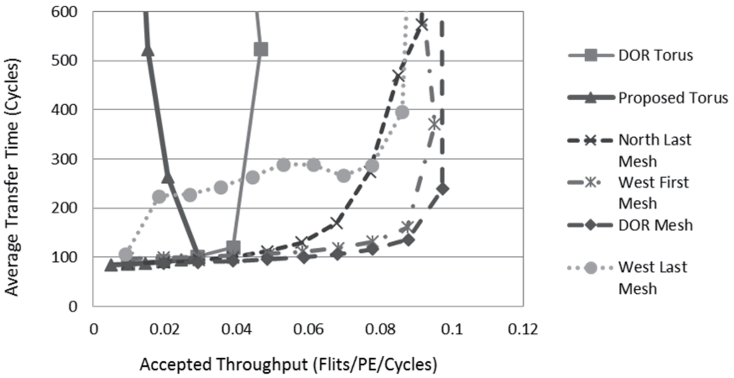


Figure 14: Results for Longest-Path Pattern

.

References

[1] J.Y. Ngai, C.L. Seitz, A framework for adaptive routing in multicomputer networks, ACM SIGARCH Computer Architecture News, Vol.19, No.1, pp.6-14, 1991.

[2] T. Schonwald, J. Zimmermann, O. Bringmann, W. Rosenstiel, Fully Adaptive Fault-Tolerant Routing Algorithm for Network-on-Chip Architectures, Digital System Design Architectures, Methods and Tools, pp.527-534, 2007.

[3] M.M. Hafizur Rahman, Yukinori Sato, Yasuyuki Miura, and Yasushi Inoguchi, Dynamic Communication Performance of a Hierarchical 3D-Torus Network, IASTED, In 10th International Conference Parallel and Distributed Computing and Networks (PDCN 2011), 2011.

[4] Y. Miura and S. Horiguchi, An Adaptive Routing for Hierarchical Interconnection Network TESH, Proc. of the Third International Conference on Parallel And Distributed Computing, Applications and Technologies, pp. 335-342, 2002.

[5] Y. Miura, Masahiro Kaneko, M.M.Hafizur Rahman and Shigeyoshi Watanabe, Adaptive Routing Algorithms and Implementation for TESH Network, Communications and Network (CN), Vol.5, No.1, pp.34-49, 2013.

[6] W.J. Dally, Performance Analysis of k-ary n-cube Interconnection Networks, IEEE Trans. on Computers, vol. 39, No.6, pp.775-785, 1990.

[7] W.J. Dally and Hiromichi Aoki, Deadlock-Free Adaptive Routing in Multicomputer Networks Using Virtual Channels, IEEE Trans. On Parallel and Distributed Systems, Vol.4, pp. 466-475, 1993.

[8] M.P.Merlin and J.P.Schweitzer, Deadlock Avoidance in Store-and-Forward Networks-1: Store and Forward Deadlock, IEEE Trans. On Communications, Vol.COM-28, No.3, pp.345-354, 1980.

[9] W.J.Dally and C.L.Seitz. Deadlock-Free Message Rouring in Multiprocessor interconnection Networks. IEEE Trans. On Computers, Vol. C-36, No.5, pp.547-553, 1987.

[10] C.S.Yang and Y. M. Tsai, Adaptive Routing in k-ary n-cube Multicomputers, Proc. of 1996 International Conference on Parallel and Distributed Systems(ICPADS'96), pp. 404-411, 1996.

[11] J.Duato, A New Theory of Deadlock-Free Adaptive Routing in Wormhole Networks, IEEE Trans. on Parallel and Distributed Systems, Vol.4, No.12, pp.1320-1331, 1993.

[12] D.H. Linder and J.C. Harden, An adaptive and fault tolerant wormhole routing strategy for k-ary n-cubes, IEEE Trans. on Computers, vol.C-40, No.1, pp.2-12, 1991.

[13] R.S. Ramanujam, Bill Lin, Destination-based adaptive routing on 2D mesh networks, 2010 ACM/IEEE Symposium onArchitectures for Networking and Communications Systems (ANCS), pp.1-12, 2010.

[14] C.J.Glass and L. M. Ni, Maximally Fully Adaptive Routing in 2D Meshes, Proc. of The 19th International Symposium on Computer Architecture, pp. 278-287, 1992.

[15] C.J.Glass, L.M.Ni, The Turn Model for Adaptive Routing, Proc. of The 25th Annual International Symposium on Computer Architecture, pp.441-450, 1998.

[16] Jie Wu, A Fault-tolerant and Deadlock-free Routing Protocol in 2D Meshes Based on Odd-even Turn Model, IEEE Trans. on Computers, Vol.52, No.9, pp.1154-1169, 2003.

[17] A.Jouraku, M.Koibuchi, H.Amano, An Effective Design of Deadlock-Free Routing Algorithms Based on 2D Turn Model for Irregular Networks, IEEE Trans. on Parallel and Distributed Systems, Vol.18, No.3, pp.320-333, 2007.

[18] W.J. Dally, Virtual-Channel Flow Control, IEEE Trans. on Parallel and Destributed Systems, Vol.3, No.2, pp.194-205, 1992.

[19] K.Matoyama, Y.Miura, and S.Watanabe, Adaptive Routing of the 2-D Torus Network, Proc.of Forum on Information Technology 2009 (FIT2009), RC-005, 2009.(In Japanese).

[20] Y.Miura, K.Shimozono, K.Matoyama, and S.Watanabe, An Adaptive Routing of the 2-D Torus Network Based on Turn Model, Proc. of 4th International Workshop on Advances in Networking and Computing, pp.587-591, 2013.12.

[21] J.Duato A necessary and Sufficient Condition for Deadlock-Free Adaptive Routing Wormhole Networks, Proc. of the International Conference on Parallel Processing, Vol.1, pp.142-149, 1994.

[22] E. Fleury and P.Fraigniaud, A General Theory for Deadlock Avoidance in Wormhole-Routing Networks, IEEE Trans. on Parallel and Distributed Systems, Vol.9, No.7, pp.626-638, 1998.